NAME

yap, more − yet another pager

SYNOPSIS

yap [**−cnuq**] [−*n*] [+*command*] [*file* ...]

DESCRIPTION

Yap is a program allowing the user to examine a continuous text one screenful at a time on a video display terminal. It does so by pausing after each screenful, waiting for the user to type a command. The commands are enumerated later. **Yap**'s main feature is, that it can page both forwards and backwards, even when reading from standard input.

The command line options are:

−*n* An integer which is the size (in lines) of a page (the initial *page-size*.

−c Normally, **yap** will display each page by beginning at the top of the screen and erasing each line just before it displays on it. If your terminal cannot erase a line, **yap** will clear the screen before it displays a page.
This avoids scrolling the screen, making it easier to read while **yap** is writing. The **-c** option causes **yap** to scroll the screen instead of beginning at the top of the screen. This is also done if your terminal cannot either erase a line or clear the screen.

−u Normally, **yap** handles underlining such as produced by nroff in a manner appropriate to the particular terminal: if the terminal can perform underlining well (t.i., the escape sequences for underlining do not occupy space on the screen), **yap** will underline underlined information in the input. The **-u** option supresses this underlining.

−n Normally, **yap** also recognises escape sequences for stand-out mode or underlining mode in the input, and knows how much space these escape sequences will occupy on the screen, so that **yap** will not fold lines erroneously. The **-n** option supresses this pattern matching.

−q This option will cause **yap** to exit only on the "quit" command.

+*command*
command is taken to be an initial command to **yap**.

Yap looks in the **YAP** environment variable to pre-set flags. For instance, if you prefer the **-c** mode of operation, just set the **YAP** environment variable to **-c**.

The commands of **yap** can be bound to sequences of keystrokes. The environment variable **YAPKEYS** may contain the bindings in the form of a list of colon-separated ‘name=sequence’ pairs. The *name* is a short mnemonic for the command, the *sequence* is the sequence of keystrokes to be typed to invoke the command. This sequence may contain a ^X escape, which means control-X, and a \X escape, which means X. The latter can be used to get the characters ‘^’, ‘\’ and ‘:’ in the sequence. There are two keymaps available, the default one and a user-defined one. You can switch between one and the other with the *change keymap* command.

The **yap** commands are described below. The mnemonics for the commands are given in parentheses. The default key sequences (if any) are given after the mnemonic. Every command takes an optional integer argument, which may be typed before the command. Some commands just ignore it. The integer argument is referred to as *i*. Usually, if *i* is not given, it defaults to 1.

visit previous file (*bf*) **P**
Visit the *i*-th previous file given in the command line.

scroll one line up or go to line (*bl*) **^K** or **k**
If *i* is not given, scroll one line up. Otherwise, *i* will be interpreted as a line number. A page starting with the line indicated will then be displayed.

bottom (*bot*) **l** or **$**
Go to the last line of the input.

display previous page (*bp*) **-**
Display the previous page, consisting of *i* lines, (or *page-size* lines if no argument is given).

display previous page and set pagesize (*bps*) **Z**

Display the previous page, consisting of *i* lines, (or *page-size* lines if no argument is given). If *i* is given, the *page-size* is set to *i*.

scroll up (*bs*) **ˆB**

Scroll up *i* lines (or *scroll-size* lines if *i* is not given. Initially, the *scroll-size* is 11).

search backwards for pattern (*bse*) **?**

Search backwards for the *i*-th occurrence of a regular expression which will be prompted for. If there are less than *i* occurrences of the expression, the position in the file remains unchanged. Otherwise, a page is displayed, starting two lines before the place where the expression was found. The user's erase and kill characters may be used to edit the expression. Erasing back past the first character cancels the search command.
Note: Some systems do not have **regex**(3). On those systems, searches are still supported, but regular expressions are not.

skip lines backwards (*bsl*) **S**

Skip *i* lines backwards and display a page.

skip pages backwards (*bsp*) **F**

Skip *i* pages backwards and display a page.

scroll up and set scrollsize (*bss*) **b**

Scroll up *i* lines (or *scroll-size* lines if *i* is not given. If *i* is given, the *scroll-size* is set to *i*.

change key map (*chm*) **X**

Change from the current key map to the other (if there is one).

exchange current page and mark (*exg*) **x**

Set the mark to the current page, and display the previously marked page.

visit next file (*ff*) **N**

Visit the *i*-th next file given in the command line.

scroll one line down or go to line (*fl*) **ˆJ** or **ˆM** or **j**

If *i* is not given, scroll one line down. Otherwise, *i* will be interpreted as a line number. A page starting with the line indicated will then be displayed.

display next page (*fp*) **<space>**

Display the next page, consisting of *i* lines, (or *page-size* lines if no argument is given).

display next page and set pagesize (*fps*) **z**

Display the next page, consisting of *i* lines, (or *page-size* lines if no argument is given). If *i* is given, the *page-size* is set to *i*.

scroll down (*fs*) **ˆD**

Scroll down *i* lines (or *scroll-size* lines if no argument is given).

search forwards for pattern (*fse*) **/**

Search forwards for the *i*-th occurrence of a regular expression which will be prompted for. If there are less than *i* occurrences of the expression, the position in the file remains unchanged. Otherwise, a page is displayed, starting two lines before the place where the expression was found. The user's erase and kill characters may be used to edit the expression. Erasing back past the first character cancels the search command.
Note: Some systems do not have **regex**(3). On those systems, searches are still supported, but regular expressions are not.

skip lines forwards (*fsl*) **s**

Skip *i* lines and display a page.

skip pages forwards (*fsp*) **f**

Skip *i* pages and display a page.

scroll down and set scrollsize (*fss*) **d**

Scroll down *i* lines (or *scroll-size* lines if *i* is not given. If *i* is given, the *scroll-size* is set to *i*.

help (*hlp*) **h**
Give a short description of all commands that are bound to a key sequence.

set a mark (*mar*) **m**
Set a mark on the current page.

repeat last search (*nse*) **n**
Search for the *i*-th occurrence of the last regular expression entered, in the direction of the last search.

repeat last search in other direction (*nsr*) **r**
Search for the *i*-th occurrence of the last regular expression entered, but in the other direction.

quit (*qui*) **Q** or **q**
Exit from **yap**.

redraw (*red*) **^L**
Redraw the current page.

repeat (*rep*) **.**
Repeat the last command. This does not always make sense, so not all commands can be repeated.

shell escape (*shl*) **!**
Invoke the shell with a command that will be prompted for. In the command, the characters '%' and '!' are replaced with the current file name and the previous shell command respectively. The sequences '\%' and '\!' are replaced by '%' and '!' respectively. The user's erase and kill characters can be used to edit the command. Erasing back past the first character cancels the command.

pipe to shell command (*pip*) **|**
Pipe the current input file into a shell command that will be prompted for. The comments given in the description of the shell escape command apply here too.

go to mark (*tom*) **'**
Display the marked page.

top (*top*) **^^**
Display a page starting with the first line of the input.

visit file (*vis*) **e**
Visit a new file. The filename will be prompted for. If you just type a return, the current file is revisited.

write input to a file (*wrf*) **w**
Write the input to a file, whose name will be prompted for.

The commands take effect immediately, i.e., it is not necessary to type a carriage return. Up to the time when the command sequence itself is given, the user may give an interrupt to cancel the command being formed.

AUTHOR

Ceriel J.H. Jacobs

SEE ALSO

regex(3).

BUGS

Yap will find your terminal very stupid and act like it, if it has no way of placing the cursor on the home position, or cannot either erase a line or insert one.

In lines longer than about 2000 characters, a linefeed is silently inserted.

The percentage, given in the prompt when **yap** reads from a file (and knows it), is not always very accurate.